Программно-аппаратные средства обеспечения информационной безопасности.

Лекция № 3

Практические методы построения изолированной программной среды.

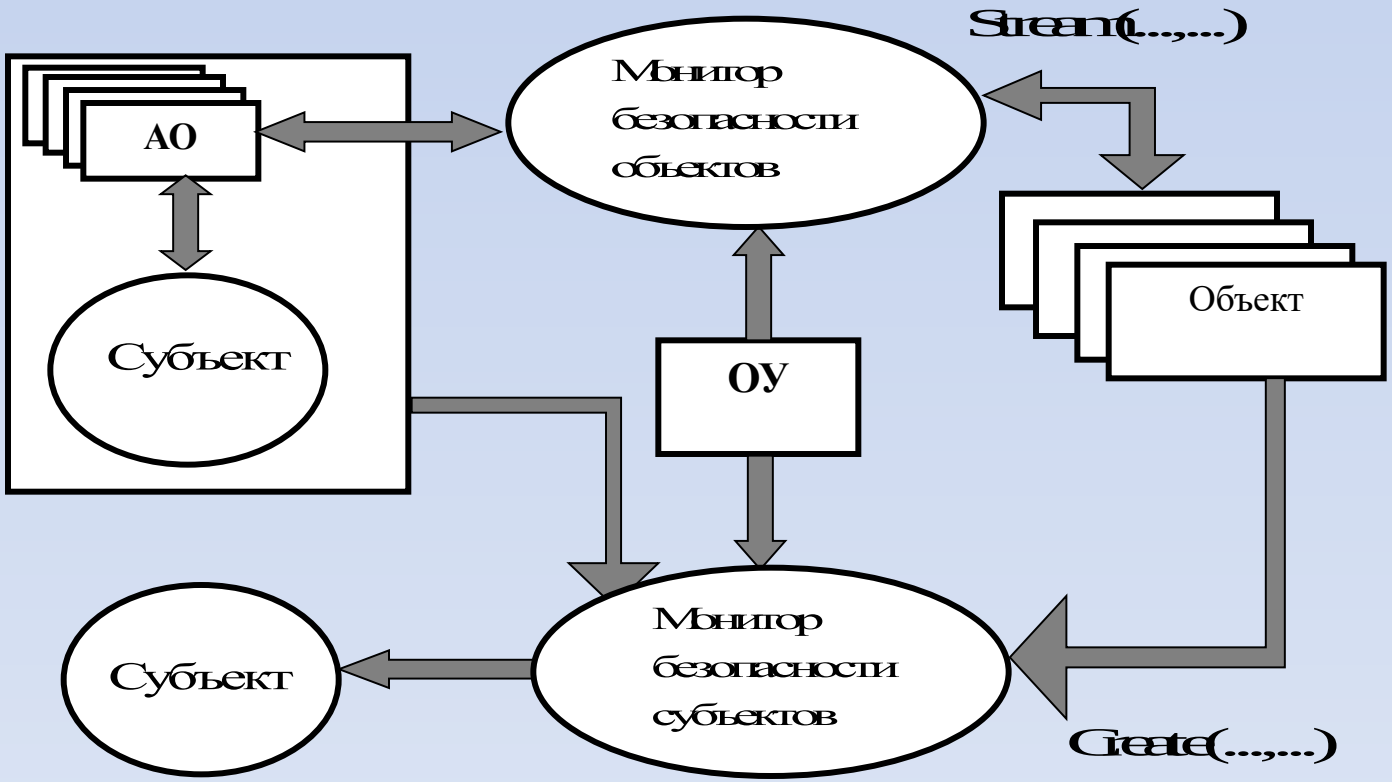
Рахманенко И.А.

МЕТОД ГЕНЕРАЦИИ ИЗОЛИРОВАННОЙ ПРОГРАММНОЙ СРЕДЫ ПРИ ПРОЕКТИРОВАНИИ МЕХАНИЗМОВ ГАРАНТИРОВАННОГО ПОДДЕРЖАНИЯ ПОЛИТИКИ БЕЗОПАСНОСТИ

- Утверждение 2 (***базовая теорема ИПС***)

  Если в момент времени $\mathbf{t_0}$ в изолированной КС действует только порождение субъектов с контролем неизменности объекта и существуют потоки от любого субъекта к любому объекту, не противоречащие условию корректности субъектов, то в любой момент времени $\mathbf{t>t_0}$ КС также остается изолированной

Ядро безопасности с учетом контроля порождения субъектов



Для выполнения ПБ необходимо

1. Убедиться в корректности субъектов
2. Убедиться в корректности любого субъекта относительно МБО и МБС.
3. Спроектировать и реализовать программно или программно-аппаратно МБС
4. Порождение любого субъекта происходит с контролем неизменности объекта-источника.
5. Реализовать МБО в рамках априорно сформулированной политики безопасности

Для выполнения ПБ необходимо

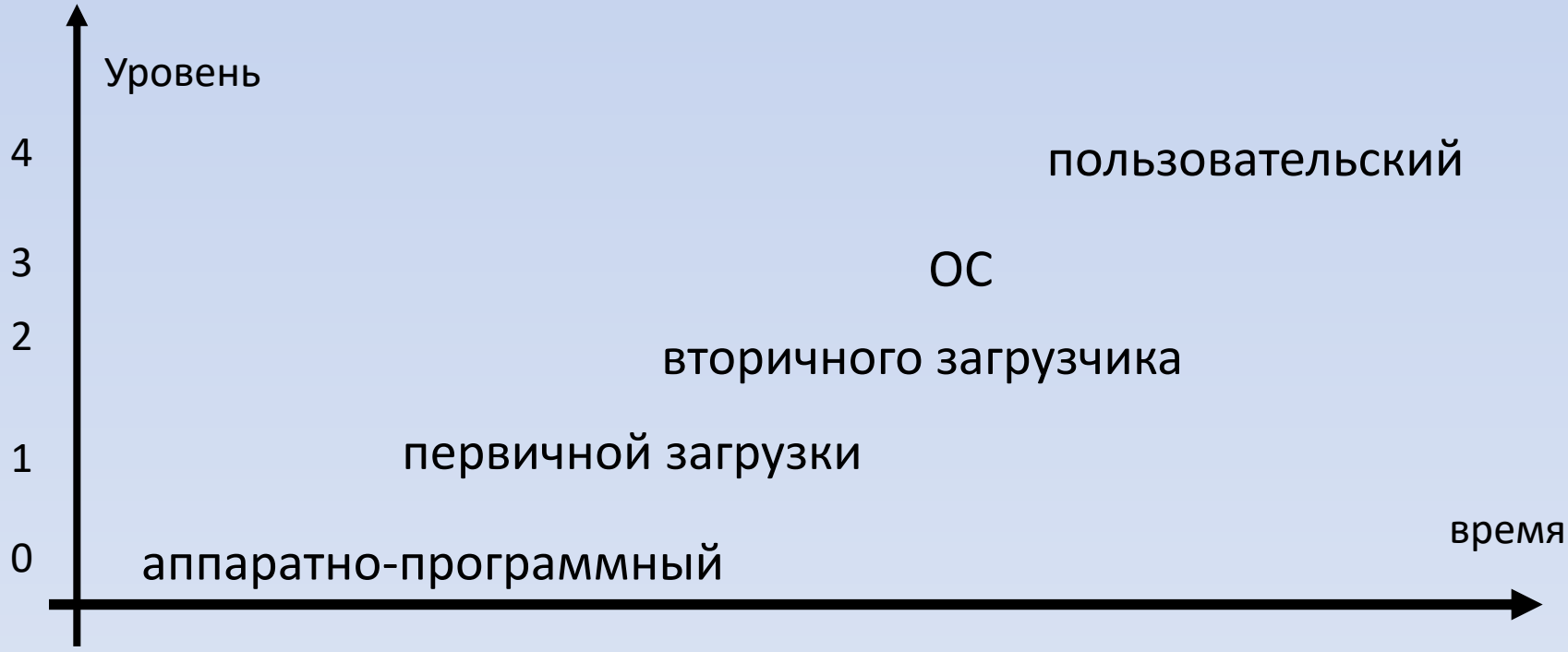
- Объект управления (ОУ) должен быть недоступен для несанкционированного изменения.
- Иначе возможно добавление к множеству разрешенных субъектов дополнительных.
- Выделенный субъект (Администратор безопасности) должен иметь возможность изменять ОУ.

Поэтапная активизация КС

- В начальные этапы активизации КС декомпозиция на субъекты и объекты динамически изменяется.

- Выделяют уровни представления информации и приведенное выше правило создания гарантированно защищенной КС применяется к каждому уровню

Иерархия уровней при загрузке ОС



Иерархия уровней при загрузке ОС

Уровень	Субъект	Локализация	Через какие функции реализуются
0	Субъект аппаратно-программного уровня	ПЗУ (Bios)	микропрограммы ПЗУ
1	Субъект уровня первичной загрузки	Загрузчик ОС	Bios или первичный загрузчик
2	Субъект уровня вторичного загрузчика (драйвер)	Драйверы ОС	Bios или первичный загрузчик
3	Субъект уровня ОС	Ядро ОС	через драйверы
4	Субъект пользовательского уровня	Прикладные программы	через ядро ОС

Иерархия уровней при загрузке ОС

- Невозможно реализовать ИПС, действующую на всех уровнях.
- Необходимо ограничивать ИПС одним уровнем.

Иерархия уровней при загрузке ОС

- Практическая реализация всех операционных систем позволяет выделить две фазы их работы:

1. Активизация субъектов с ростом уровня представления объектов (фаза загрузки или начальная фаза)
2. Фаза стационарного состояния (когда уровень представления объектов не увеличивается

Стационарное состояние

- Пусть в КС выделяется конечное число уровней представления объектов U={0,…,R}, где R – максимальный уровень представления объекта.
- Стационарное состояния КС - в отображениях ***Stream*** и ***Create*** участвуют только объекты уровня R.
- Например запрещаются операции ниже уровня "файл" со стороны субъектов прикладного уровня

Этапы работы ОС

- активизация субъектов с ростом уровня представления объектов (фаза загрузки или начальная фаза)
- фаза стационарного состояния (когда уровень представления объектов не увеличивается).

Иерархия уровней при загрузке ОС

- Практическая реализация ИПС состоять из двух этапов:

1. Предопределенное выполнение начальной фазы, включающее в себя момент активизации МБС и МБО
2. работа в стационарной фазе в режиме ИПС, с контролем неизменности объектов-источников

Обозначения

- Z_L – последовательность пар $(i,j)_t$ (t=0,1,2,…,**l**-1 – моменты времени) длины **l**, такие, что ***Create(Si,Oj)***[t]->**Sm**[t+**l**].
- Sz – множество всех субъектов, включенных в последовательность Z_L;
- O_Z – множество всех объектов, включенных в последовательность Z_L.

Утверждение 3. (Условие одинакового состояния КС)

Состояние КС в моменты времени tx1 и tx2 (tx1 и tx2 исчисляются для двух отрезков активности КС от нулевого момента активизации КС to1 и to2 – например, включения питания аппаратной части) одинаково, если:

1. tx1=tx2,
2. тождественны субъекты Si[to1] и Si[to2],
3. неизменны все объекты из множества O_Z,
4. неизменна последовательность Z_L.

Утверждение 4 (достаточное условие ИПС при ступенчатой загрузке)

- При условии неизменности Z_L и неизменности объектов из O_Z в КС с момента времени установления неизменности Z_L и O_Z действует ИПС.

Утверждение 5 (требования к субъектному наполнению ИПС)

Для того, чтобы ИПС поддерживалась в течение всего времени активности КС, достаточно, чтобы в составе программного обеспечения, могущего быть инициированным в ИПС

не было функций порождения субъектов и прекращения их работы, кроме заранее предопределенных при реализации МБС

не существовало возможностей влияния на среду выполнения (под средой выполнения понимается множество ассоциированных объектов) любого процесса

не существовало возможностей инициирования потоков к объектам логического уровня менее R.

ФОРМИРОВАНИЕ И ПОДДЕРЖКА ИЗОЛИРОВАННОЙ ПРОГРАММНОЙ СРЕДЫ

Формирование и поддержка ИПС

- Предположим, что в КС работают N субъектов-пользователей, каждый i-й из которых характеризуется некоторой персональной информацией Ki, не известной другим пользователям и хранящейся на некотором материальном носителе.
- Существует выделенный субъект - администратор системы, который знает все Ki.

Формирование и поддержка ИПС

- Администратор КС присваивает i-му пользователю полномочия, заключающиеся в возможности исполнения им только заданного подмножества программ, Ti={Pi1, Pi2, ... Pit}.

Несанкционированный доступ

- Использование имеющихся на жестком диске ЭВМ программ либо субъектом, не входящим в N допущенных, либо i-м пользователем вне подмножества своих полномочий Ti.
- Субъект, пытающийся проделать данные действия, называется злоумышленником.

Виды НСД

- Непосредственный НСД
- Опосредованный НСД

Непосредственный НСД

- Злоумышленник, используя некоторое ПО пытается непосредственно осуществить операции чтения или записи (изменения) интересующей его информации.
- Если предположить, что в Ti нет программ, дающих возможность произвести НСД (это гарантирует администратор при установке полномочий), то НСД может быть произведен только при запуске программ, не входящих в Ti.

Опосредованный НСД

- Обусловлен общностью ресурсов пользователей и заключается во влиянии на работу другого пользователя через используемые им программы (после предварительного изменения их содержания или их состава злоумышленником).
- Программы, участвующие в опосредованном НСД - разрушающие программные воздействия (РПВ) или программными закладками.

РПВ

- Могут быть внедрены i-м пользователем в ПО, принадлежащее j-му пользователю только путем изменения программ, входящих в Tj.

Предотвращение РПВ

- Система защиты от НСД должна обеспечивать контроль за запуском программ, проверку их целостности и активизироваться всегда для любого пользователя.
- Выполнение контроля целостности и контроля запусков ведется на основе Ki для каждого пользователя.

Защитный механизм должен

- В некоторый начальный момент времени требовать у субъекта предъявления аутентифицирующей информации и по ней однозначно определять субъекта и его полномочия Ti;
- В течении всего времени работы пользователя i должен обеспечивать выполнение программ только из подмножества Ti;
- Пользователь не должен иметь возможности изменить подмножество Ti и/или исключить из дальнейшей работы защитный механизм и его отдельные части.

Предположения

- В ПЗУ (BIOS) и в ОС (в том числе и в сетевом ПО) отсутствуют специально интегрированные в них возможности НСД.
- Пользователь работает с программой, в которой также исключено наличие каких-либо скрытых возможностей (проверенные программы).

НСД возможен

1. Проверенные программы используются на другой ЭВМ с другим BIOS.
2. Проверенные программы будут использованы в аналогичной, но не проверенной ОС.
3. Проверенные программы используются на проверенной ЭВМ и в проверенной ОС, но запускаются еще и не проверенные программы, потенциально несущие в себе возможности НСД.

НСД гарантированно невозможен

1. На ЭВМ с проверенным BIOS установлена проверенная ОС.
2. Достоверно установлена неизменность ОС и BIOS для данного сеанса работы.
3. Кроме проверенных программ в данной программно-аппаратной среде не запускалось и не запускается никаких иных программ, проверенные программы перед запуском контролируются на целостность.
4. Исключен запуск проверенных программ в какой-либо иной ситуации, т.е. вне проверенной среды.
5. Условия 1-4 выполняются в любой момент времени для всех пользователей, аутентифицированных защитным механизмом.

ИПС

- При выполнении всех условий программная среда называется изолированной.
- От системы защиты требуется:

1. Невозможность запуска программ помимо контролируемых ИПС событий.

2. Отсутствие в базовом ПО возможностей влиять на среду функционирования уже запущенных программ (требование невозможности редактирования оперативной памяти).

Контроль целостности

- Основной элемент поддержания изолированности среды - контроль целостности.

РЕАЛИЗАЦИЯ ИПС С ИСПОЛЬЗОВАНИЕМ МЕХАНИЗМА РАСШИРЕНИЯ BIOS

Условия

- 2 этапа - этап установки ИПС и этап эксплуатации ИПС.
- N пользователей, каждый i-й имеет персональную информацию Ki, не известную другим пользователям и хранящуюся на материальном носителе (например, устройстве Touch Memory).
- Администратор КС - знает все Ki и проводит этап установки.
- Пользователи участвуют в этапе эксплуатации.

Установка ИПС

1. В ПЭВМ устанавливается плата, реализует:

- чтение Ki,
- идентификацию пользователя i по Ki,
- чтение множества доступных для выполнения пользователем i задач Pi1, Pi2, ..., Pim, и информации Mi1, Mi2, ... Mim, фиксирующей целостность файлов Fi1, ..., Fim каждой задачи.

Установка ИПС

2. Администратор определяет для пользователя i набор задач и соответствующих задачам исполняемых файлов

$$\{Pit, Fit\},\ t=1,...,mi;\ i=1, ..., N,$$

где mi - число разрешенных к запуску задач для i-го пользователя.

Установка ИПС

3. Администратор формирует (и заносит на носитель) или считывает с носителя для i-го пользователя его Ki и вычисляет значения для последующего контроля целостности

$$Mir=f(Ki, Fir, Pir),$$

где f - функция фиксации целостности (хэш-функция).

Установка ИПС

4. Администратор проделывает действия 2 и 3 для всех N пользователей.

5. Администратор устанавливает в ПС модуль активизации ИПС и фиксирует его целостность.

Фиксируется также целостность файлов ОС Foc.

Эксплуатация ИПС

1. Включение питания и активизация расширенного BIOS:

а) идентификация пользователя по его Ki (при успехе п. б);

б) проверка целостности всех включенных в ЭВМ BIOS (при положительном исходе п. в);

в) чтение по секторам файлов ОС и проверка их целостности;

г) чтение как файлов Рипс (с помощью функций ОС) и проверка целостности;

д) активизация сетевого ПО;

е) активизация процесса контроля Рипс;

ж) запуск избранной задачи i-го пользователя.

Эксплуатация ИПС

2. Работа в ИПС. Запуск процесса Ps сопровождается проверками:

а) принадлежит ли Fs к множеству разрешенных для i (Ti), если да, то п. б), иначе запуск игнорируется;

б) совпадает ли G=f(Ki,Fs,Ps) с M=f(Ki,Fs,Ps), вычисленной администратором;

в) при положительном исходе проверки б) задача запускается.

НСД невозможен

- Пункты 1-5 выполнены (слайд 31).
- Пункт 1 гарантируется при установке системы администратором.
- Пункты 2, 4 и 5 обеспечиваются платой (загрузка со съемного носителя невозможна; пользователь допускается к работе только при проверке Ki).
- Пункт 3 реализован программным модулем контроля запусков и контроля целостности задач, входящим в состав ИПС.

UEFI

Что такое UEFI?

- UEFI (Unified Extensible Firmware Interface — единый расширяемый интерфейс встроенного ПО)
- Интерфейс между операционной системой и микропрограммами, управляющими низкоуровневыми функциями оборудования
- Назначение: корректно инициализировать оборудование при включении системы и передать управление загрузчику операционной системы

Что такое UEFI?

- Цель UEFI заключается в определении стандартного способа взаимодействия операционной системы со встроенным ПО платформы во время процесса загрузки.
- До появления UEFI основным механизмом взаимодействия с оборудованием во время процесса загрузки были программные прерывания.

Что такое UEFI?

- UEFI позволяет реализовать модульную структуру встроенного ПО, которая предоставляет разработчикам оборудования и систем значительную гибкость в разработке встроенного ПО для более требовательных современных вычислительных сред

Безопасная загрузка

- UEFI имеет процесс проверки встроенного ПО, который называется безопасной загрузкой (Secure boot).
- Безопасная загрузка определяет, как встроенное ПО платформы управляет сертификатами безопасности, проверкой встроенного ПО и реализацией интерфейса (протокола) между встроенным ПО и операционной системой.

Безопасная загрузка

- Защита от вредоносных загрузчиков.
- До запуска операционной системы разрешен запуск только подписанных и сертифицированных "известных" загрузчиков и кода

Безопасная загрузка

- Среда, предшествующая загрузке ОС, уязвима для атак, осуществляемых с помощью передачи функций загрузчика вредоносным загрузчикам.
- Эти загрузчики остаются необнаруженными мерами безопасности операционной системы и антивирусным ПО

BIOS - UEFI

- Возможна загрузка вредоносного ПО

- Использование загрузчика с ЭП

Как работает UEFI

- При включении питания компьютера запускается процесс выполнения кода, который настраивает процессор, память и периферийные устройства, подготавливая их к выполнению операционной системы.
- Этот процесс выполняется одинаково для всех платформ независимо от архитектур, на которых они основаны (x86, ARM и т. д.).

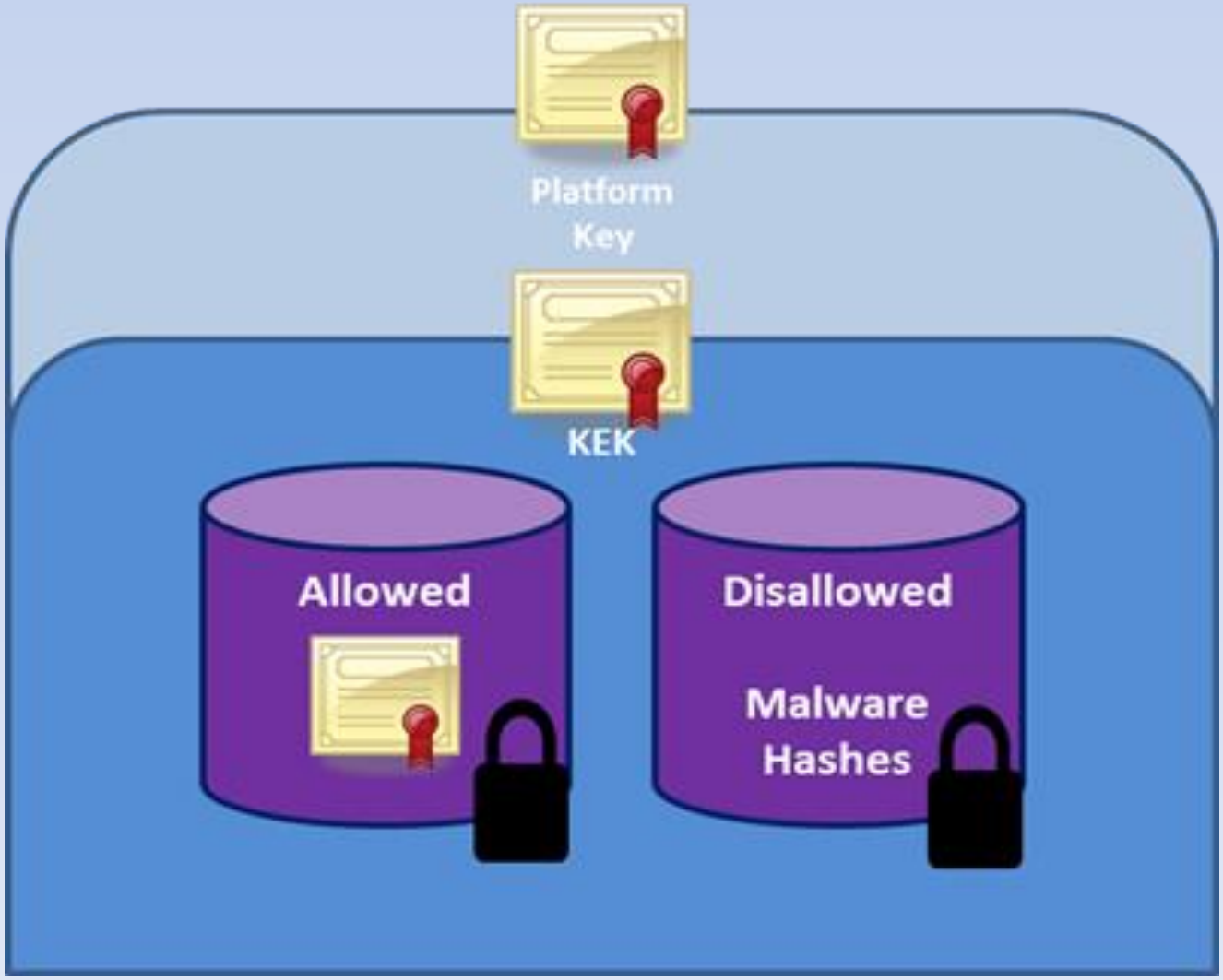
Как работает UEFI

- Проверяется сигнатура кода встроенного ПО, присутствующего на периферийных устройствах, таких как сетевые карты, запоминающие устройства или видеоадаптеры.
- Этот код на устройствах, называемый дополнительными ПЗУ, продолжает процедуру настройки, обеспечивая подготовку периферийных устройств к передаче управления ОС.

Как работает UEFI

- Встроенное ПО проверяет внедренную сигнатуру
- Если эта сигнатура присутствует в БД сигнатур, то этот модуль получает разрешение на выполнение.
- БД определяет, может ли продолжиться процесс загрузки

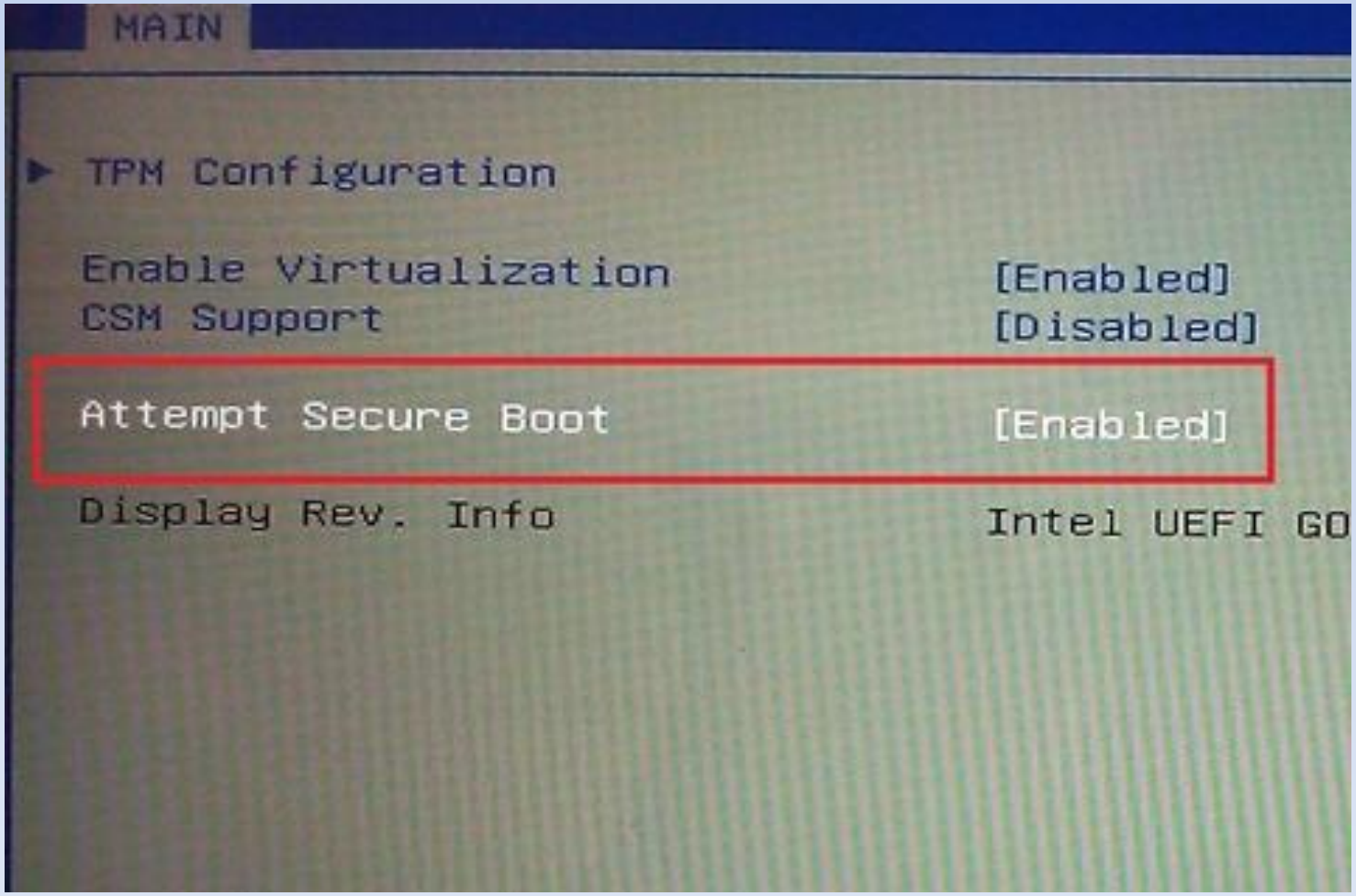
Как работает UEFI



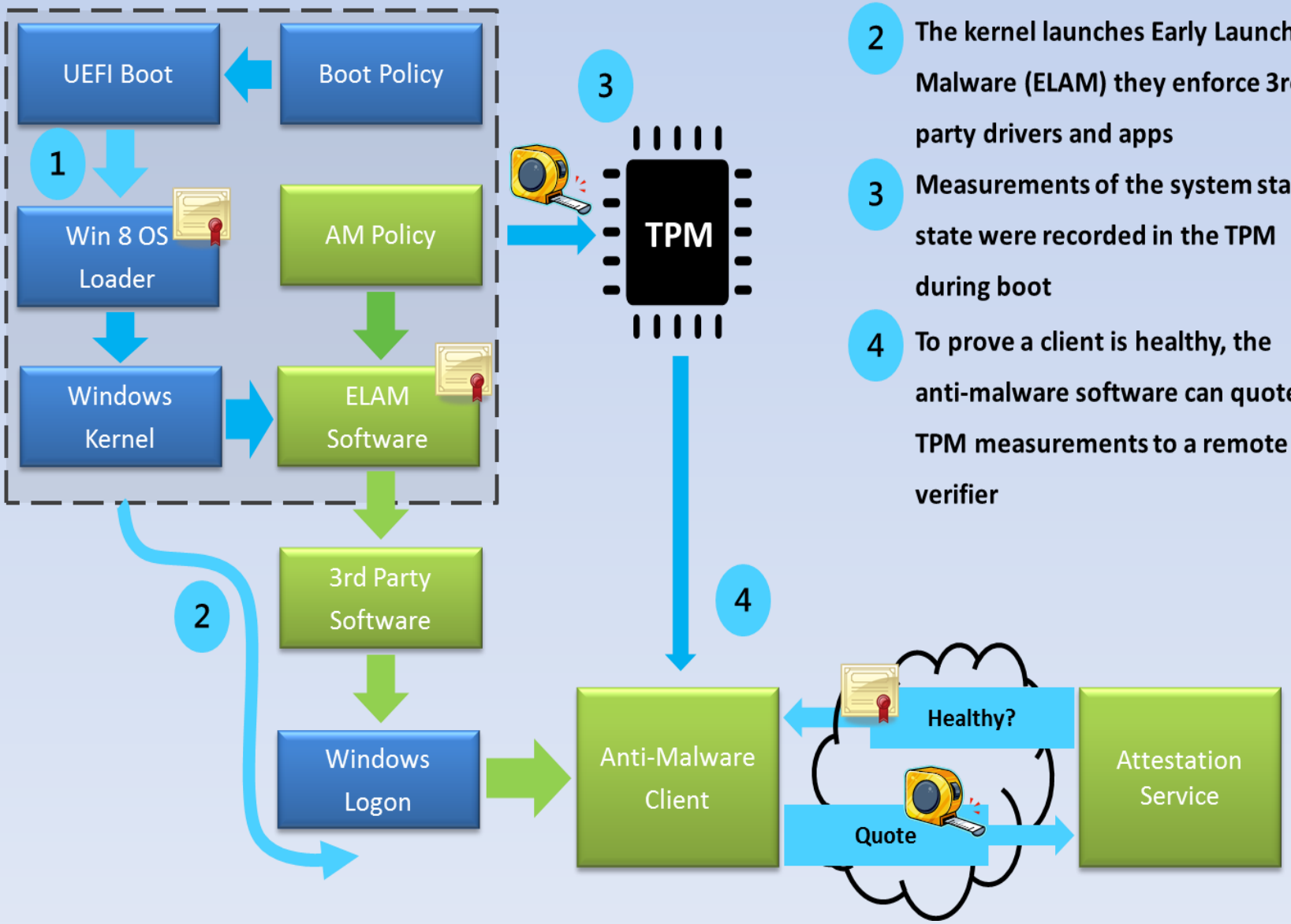
Как работает UEFI

- БД с разрешенными загрузчиками содержит хэши надежного ПО и загрузчиков ОС.
- Другая БД содержит хэши вредоносных программ.

Возможность отключения безопасной загрузки



Windows 8 Platform Integrity Architecture



1. Secure boot (UEFI) prevents running a unknown OS loader
2. The kernel launches Early Launch Anti-Malware (ELAM) they enforce 3rd party drivers and apps
3. Measurements of the system start state were recorded in the TPM during boot
4. To prove a client is healthy, the anti-malware software can quote TPM measurements to a remote verifier

Рассмотренные вопросы

- Метод генерации изолированной программной среды при проектировании механизмов гарантированного поддержания политики безопасности
- Реализация ИПС с использованием механизма расширения BIOS
- UEFI